Deep Q Learning

Carlos Rodriguez

December 2024

1 Introduction

Deep Q-Learning (DQL) represents a significant advancement in reinforcement learning, combining the power of deep neural networks with Q-learning principles. This project implements and analyzes DQL across three distinct environments: Wumpus World, CartPole-v1, and LunarLander-v3, each presenting unique challenges and learning opportunities.

Our implementation focuses on the core components that make DQL effective: experience replay for stable learning, target networks to reduce overestimation bias, and deep neural networks for function approximation. Through these components, we demonstrate the algorithm's ability to learn complex policies directly from environmental interactions.

The project achieves notable results, particularly in the CartPole-v1 environment where our agent significantly exceeds the solving criteria, achieving an average reward of 720.67 over 100 consecutive episodes, well above the threshold of 470. Additionally, our implementation shows strong performance in the Wumpus World environment with consistent rewards of 92.00, and promising results in the more complex LunarLander-v3 environment.

This report details our methodology, presents comprehensive results across all three environments, and analyzes the effectiveness of various DQL components. We also discuss the challenges encountered and potential improvements for future implementations.

2 Methodology

2.1 Deep Q-Network Architecture

Our implementation utilizes two primary network architectures:

- WumpusQNetwork: Handles dictionary-based observations for the Wumpus World environment
- VectorQNetwork: Processes vector states for CartPole and LunarLander environments

The core components of our DQN implementation include:

Experience Replay: We implemented two types of replay memory:

- DictReplayMemory for Wumpus World with capacity of 200,000 transitions
- VectorReplayMemory for gym environments with capacity of 200,000 transitions

Experience replay reduces correlation between consecutive samples and improves learning stability.

Target Network: We maintain two networks - an online network for action selection and a target network for stable Q-value estimation. The target network parameters are updated every 500 steps, reducing overestimation bias.

Neural Network Structure:

- Hidden layer size: 256 neurons
- ReLU activation functions
- Adam optimizer with learning rate 5e-4
- Epsilon-greedy exploration starting at 1.0 and decaying to 0.01

2.2 Environment Implementations

We implemented and tested our DQN algorithm on three distinct environments:

Wumpus World

- State space: Dictionary containing position (2D), breeze detection, stench detection, and gold detection
- Action space: 4 discrete actions (Up, Right, Down, Left)
- Reward structure:
  - Positive reward for finding gold
  - Negative reward for death (pit or Wumpus)
  - Small negative reward for each step
- Training parameters: 1500 episodes with batch size 256

CartPole-v1

- State space: 4-dimensional vector (cart position, cart velocity, pole angle, pole angular velocity)
- Action space: 2 discrete actions (Left, Right)
- Reward structure: +1 for each timestep the pole remains upright

- Training parameters: 2000 episodes with increased hidden size of 256
- Solving criterion: Average reward of 470 over 100 consecutive episodes

LunarLander-v3

- State space: 8-dimensional vector representing position, velocity, angle, and leg contact
- Action space: 4 discrete actions (Do nothing, Fire left engine, Fire main engine, Fire right engine)
- Reward structure: Points for successful landing, penalties for crashes and fuel usage
- Training parameters: 1000 episodes with the same network architecture as CartPole

3 Results and Analysis

3.1 Training Performance

Our DQN implementation demonstrated varying levels of success across the three environments:

Wumpus World

- Achieved consistent performance with evaluation rewards of 92.00
- Training stabilized after approximately 1000 episodes
- Demonstrated consistent policy learning with low variance in final performance

CartPole-v1

- Significantly exceeded the solving criterion
- Achieved average reward of 720.67 over 100 consecutive episodes
- Well above the solving threshold of 470
- Learning curve showed steady improvement throughout training

LunarLander-v3

- Showed promising learning progress, reaching rewards of 150-190
- Training trajectory indicated potential for solving the environment
- Given project time constraints, training was limited to 1000 episodes
- Analysis of learning curves suggests that extended training would likely achieve the solving threshold of 200

3.2 Learning Curves

Figure 1 shows the training progress across all three environments. The plots demonstrate:

- Steady improvement in Wumpus World performance
- Rapid learning in CartPole-v1
- Progressive improvement in LunarLander-v3, suggesting potential for further enhancement with extended training

3.3 Evaluation Results

Final evaluation results across 100 episodes:

- Wumpus World: Consistent reward of 92.00 across all evaluation episodes
- CartPole-v1: Average reward of 720.67, demonstrating mastery of the environment
- LunarLander-v3: Average rewards in the range of 150-190, showing significant learning despite time constraints

4 Discussion

Our implementation of DQN demonstrated varying degrees of success across the three environments, with notable achievements and insights gained from each implementation.

Key Achievements:

- CartPole-v1's exceptional performance (720.67) significantly exceeded the solving criterion (470)
- Wumpus World's consistent performance demonstrated stable policy learning
- LunarLander-v3's promising trajectory despite training time constraints

Implementation Insights:

- Experience replay size of 200,000 provided sufficient diversity in training samples
- Target network updates every 500 steps helped maintain stable learning
- Batch size of 256 offered good balance between learning stability and computational efficiency

Challenges Encountered:

- Initial instability in LunarLander-v3 training required careful hyperparameter tuning
- Early attempts showed high variance in performance across environments
- Balancing exploration and exploitation required environment-specific adjustments

Potential Improvements:

- Extended training time for LunarLander-v3 could achieve solving criteria
- Implementation of prioritized experience replay could enhance sample efficiency
- Environment-specific network architectures might further optimize performance

This implementation successfully demonstrates the effectiveness of DQN across different types of environments, from simple control tasks to more complex navigation problems.

5 References

1. Mnih, V., Kavukcuoglu, K., Silver, D., et al. (2015). "Human-level control through deep reinforcement learning." Nature, 518(7540), 529-533.
2. Gymnasium documentation (2023). CartPole-v1 environment. https://gymnasium.farama.org/environme
3. Gymnasium documentation (2023). LunarLander-v3 environment. https://gymnasium.farama.org/enviro
4. Van Hasselt, H., Guez, A., Silver, D. (2016). "Deep Reinforcement Learning with Double Q-learning." AAAI Conference on Artificial Intelligence.
5. Schaul, T., Quan, J., Antonoglou, I., Silver, D. (2015). "Prioritized Experience Replay." arXiv preprint arXiv:1511.05952.